Ocean Engineering, Volume 161, 1 August 2018, Pages 308-321
DOI: 10.1016/j.oceaneng.2018.01.025

An energy-efficient path planning algorithm for unmanned surface vehicles

Hanlin Niu[a], Yu Lu[a], Al Savvaris[a,*], Antonios Tsourdos[a]

[a]*School of Aerospace, Transport and Manufacturing, Cranfield University, Cranfield, MK43 0AL United Kingdom*

Abstract

The sea current state affects the energy consumption of Unmanned Surface Vehicles (USVs) significantly and the path planning approach plays an important role in determining how long the USV can travel. To improve the endurance of the USV, an energy efficient path planning approach for computing feasible paths for USVs that takes the energy consumption into account based on sea current data is proposed. The approach also ensures that the USV remains at a user-configurable safety distance away from all islands and coastlines. In the proposed approach, Voronoi diagram, Visibility graph, Dijkstra's search and energy consumption function are combined, which allows USVs to avoid obstacles while at the same time using minimum amount of energy. The Voronoi-Visibility (VV) energy-efficient path and the corresponding shortest path were simulated and compared for ten missions in Singapore Strait and five missions for islands off the coast of Croatia. Impact of parameters such as mission time, the USV speed and sea current state on the results were analysed. It is shown that the proposed VV algorithm improves the quality of the Voronoi energy efficient path while keeping the same level of computational efficiency as that of the Voronoi energy efficient path planning algorithm.

Keywords:
Unmanned surface vehicles, Energy efficient, Path planning, Voronoi diagram, Visibility graph, Dijkstra's search

1. Introduction

Over the past decade, advances in automation technologies and navigation equipment accelerated the development and the range of missions USVs can undertake. USVs can be defined as unmanned vehicles which perform tasks in a variety of cluttered environments without any human intervention, and there is a great interest of potentially using USVs in a wide range of potential applications. USVs can be deployed to carry out a variety of tasks, such as scientific research, environmental missions, ocean resource exploration, military uses, and other applications (Breivik et al., 2008; Naeem et al., 2008; Sutton et al., 2011; Bingham et al., 2012). However, the limited energy capacity of USVs limits the range and duration during their deployment. Path planning is a critical part of the USVs system, which determines the level of autonomy of the USV. Path planning algorithm aims to optimize the safety, energy consumption or travelling time. In the sense of saving energy, the effect of the sea current in the path planning was presented in (Garau et al., 2005) and an A* search algorithm with time optimal cost was proposed. A* algorithm was also applied to find a path for an AUV to navigate in the presence of currents (Koay and Chitre, 2013) and obstacles while consuming the minimum amount of energy. However, the final paths are not optimal and have redundant turns because the searching space was limited by the grid cell. An energy efficient path planning algorithm based on EEA* was proposed in (Lee et al., 2015). The proposed algorithm used a realistic energy cost considering the tidal current and water-depth. The proposed algorithm was finally compared with a classical distance based A* algorithm. In this approach, the collision free space was constructed using exact cell decomposition approach. The cell decomposition method uses nonoverlapping cells to represent the free-space (C_f) connectivity. The cell decomposition method (Noborio et al., 1990) includes exact decomposition method (Avnaim et al., 1988) and approximate decomposition method. The cell decomposition method, although simple to implement, seldom

*Corresponding author
Email addresses: h.niu@cranfield.ac.uk (Hanlin Niu), yu.lu.2@cranfield.ac.uk (Yu Lu), a.savvaris@cranfield.ac.uk (Al Savvaris), a.tsourdos@cranfield.ac.uk (Antonios Tsourdos)

yields high-quality paths (Bhattacharya and Gavrilova, 2008). The exact decomposition method discretizes the free space recursively, stopping when a cell is entirely in free-space or entirely inside an obstacle, otherwise, the cell is further divided. The exact cell decomposition techniques are faster than the approximate one, but the path is not optimal. The decomposition method can yield near-optimal paths by increasing the grid resolution, but the computational time will increase drastically.

The level set method was used to solve the environmental influence problem for AUV path planning (Agarwal and Lermusiaux, 2011). The Anisotropic Fast Marching (AFM) method was applied to address similar problems but in an environment where relatively stronger currents exist (Petres et al., 2005) . The AFM is an improved version of the FM method, which has higher computational efficiency than the level set method (Agarwal and Lermusiaux, 2011). Also, the optimal collision free path generated by the AFM is able to provide the guaranteed convergence, which has been intuitively explained in (Konukoglu et al., 2007) and mathematically proven in (Mirebeau, 2014). However, these studies have only been applied on AUV platforms, which has limited constraints than the USV environment. In the USV environment, additional constraints such as wind, tidal currents and COLREGS regulations, should be taken into account, for which the conventional AFM cannot implement. A multi-layered fast marching (MFM) method for USV path planning was proposed in (Song et al., 2017) to address these problems. The MFM method took into account of the sea current data and generated path in the collision free space, which was constructed by potential field method. The potential method used an attractive/repulsive vector field to construct the collision free space around the obstacles (Warren, 1989).

The cell decomposition method or potential field method have been applied to yield the collision free space in dealing with USV energy efficient path planning problem, little work about roadmap based energy efficient path planning method has been done. The roadmap method attempts to capture the free-space connectivity with a graph. The sampling-based roadmap method includes probabilistic roadmap method (Amato and Wu, 1996) (Kavraki and Latombe, 1998), rapidly exploring random tree (Kuffner and Latombe, 2000) (Kuffner and LaValle, 2000), expansive space planner (Hsu et al., 1997) and random walk planner (Carpin and Pillonetto, 2005). Some other well known roadmap based approaches based on computational geometry include Visibility graph and Voronoi diagram (Pehlivanoglu, 2012) (Candeloro et al., 2017). The illustration of Voronoi diagram and Visibility graph are shown in Fig. 1 and Fig. 2, respectively. The Voronoi roadmap can be used as the environment model that is more computational efficient and better adapted to the context of mobile robots than regular grids (Benavides et al., 2011). The advantage of using Voronoi diagram as a roadmap, among which the Visibility graph prevails, is its efficiency. The Voronoi diagram can be constructed in just $O(nlog(n))$ time, where n is the number of the vertices. The fastest known algorithm for constructing Visibility graph takes $O(n^2)$ time (Warren, 1989) (Ghosh and Mount, 1991) and it has $O(n^2)$ edges in the worst case, which makes it impractical for large spatial dataset. Since Voronoi diagram has $O(n)$ edges, searching a Voronoi diagram based roadmap is much faster than searching a visibility graph. Another advantage of Voronoi diagram is that the constructed roadmap will keep the path away from the obstacles as far as possible, while Visibility graph will keep the path as near as possible to the obstacles, since the Visibility graph uses the edges of the obstacles as possible paths. The role of the Voronoi diagram approach in path planning is shown in (Marbate and Jaini, 2013). The disadvantage of the Voronoi diagram is that the generated path may be far from optimal. Therefore, to take the advantage of the computational efficiency of the Voronoi diagram, the generated path needs to be refined.

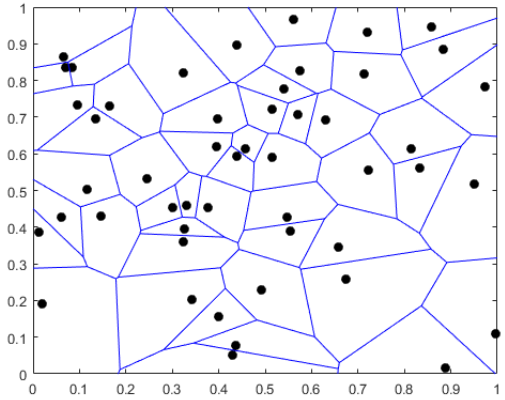


Fig. 1. Voronoi Diagram: the blue lines divide the plane into different cells. In each cell, all of the points are closer to the black point within the cell than the black points outside of the cell. The edges of two adjacent regions are composed of points equidistant from the two given points. Therefore, the set of lines equidistant from multiple points form the Voronoi diagram.

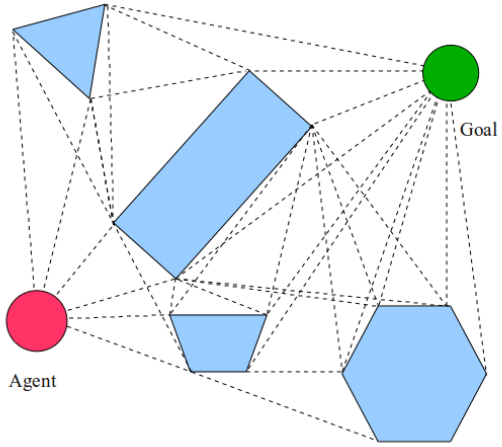


Fig. 2. Visibility Graph: the dashed lines represent candidate paths for the vehicle and the polygons represent the obstacles

Numerical optimization techniques has been applied to refine the path obtained from a roadmap (Kim et al., 2003). The edges those were nearer to the obstacles were assigned higher costs. The Dijkstra's search algorithm was applied to search for the path with minimum cost. However, there is no guarantee that the method will generate an optimal path, as the path is constrained to the edges in the roadmap. A B-Spline approximation method has been used to improve the smoothness of the path obtained from the roadmap (Ibarra-Zannatha et al., 1994). In the work of Wein et al. (2005), a new diagram called $VV^{(C)}$ diagram was proposed. The $VV^{(C)}$ diagram integrates Visibility graph and Voronoi diagram and can also keep the path away from the obstacle with clearance distance c, which is a user-configurable value. However, this algorithm is Visibility graph based, the processing time is $O(n^2log(n))$, which is impractical for large spatial datasets. In (Masehian and Amin-Naseri, 2004), the Voronoi diagram, Visibility graph and potential field were integrated into a single architecture to provide a parametric tradeoff between the safest and shortest path and the generated paths are shorter than then Voronoi and potential field methods, and faster than the Visibility graph. However, this algorithm is fairly complicated, and although the path length is shorter than those generated by the Voronoi diagram and potential field method, it still contains bumps and rudimentary turns. A Voronoi diagram based shortest path planning algorithm was proposed in (Bhattacharya and Gavrilova, 2008), the generated Voronoi shortest path was refined by minimizing the number of the waypoints and the final path was finally smoothed by using corner-cutting technique. It was demonstrated that the proposed algorithm reduced the length of the Voronoi path and still kept the computational efficiency $O(nlog(n))$.

In this research, the Voronoi-Visibility energy efficient path planning algorithm is proposed. The proposed algorithm integrates the computational efficiency advantage of the Voronoi diagram and the optimal advantage of the Visibility graph. It improves the quality of the Voronoi path and keeps the same level of the computational efficiency as that of the Voronoi diagram method, which makes it practical to process the large spatial dataset. The proposed algorithm not only minimize the energy consumption also ensures the safety of the USV by keeping the USV a configurable clearance distance from the coastlines. The rest of this paper is organized as follows: Section 2 presents the problem statement of developing the energy efficient path planning algorithm. The methodology is introduced in Section 3. Section 4 presents the numerical simulation. The conclusion and future work are given in Section 5.

2. Problem Statement

The USV energy efficient path planning algorithm for long range mission includes three challenges: Firstly, in processing with the large spatial dataset, it should be computational efficient. Secondly, the generated path should keep the USV a clearance distance from the island coastline that will keep the safety of the USV in the existence of the map inaccuracy. Thirdly, the path planning approach should take account of the spatially variant sea current data and generate an energy efficient path.

2.1. Large Spatial Dataset

For a long range USV mission scenario, the high-resolution spatial dataset will improve the accuracy of the result, but will also cause a computational burden. For instance, there are 98 islands in Singapore and 4128 vertices are used for representing Singapore islands, as shown in Fig. 3. Even using fastest Visibility graph to construct the roadmap, the construction complex will be $O(n^2)$ and edges number will be $O(n^2)$ in the worst case, which renders it impractical for large spatial dataset. Although Voronoi diagram can generate the roadmap in $O(nlog(n))$ time, its path is far from optimal. Therefore, a method to combine both the advantage of the Voronoi diagram and Visibility graph is necessary.

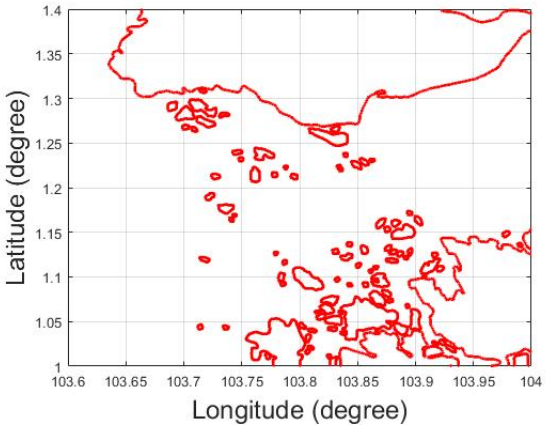


Fig. 3. Singapore islands

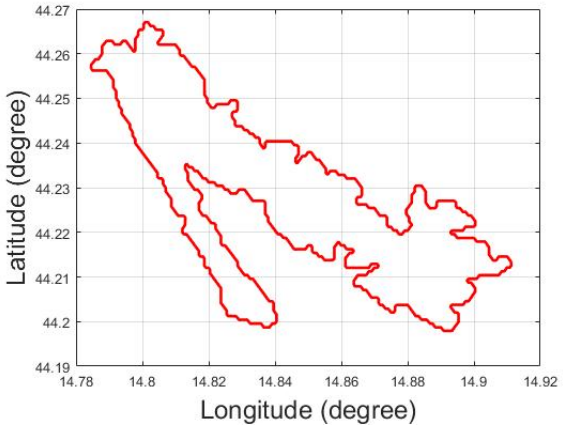


Fig. 4. Croatian island

2.2. Clearance Distance c

The generated path should keep a clearance distance from the island coastlines. The sea area near to the coastline may have crowded traffic that is inconvenient for travelling or shallow area that is dangerous to the USV. Moreover, the map data may be inaccurate in some cases. For instance, a Croatian island is plotted in Fig. 4 and the data is achieved from the Global Self-Consistent Hierarchical High Resolution Shorelines (GSHHS) dataset). However, when the island profile is mapped into google map in Fig. 5, the gap between two profiles from different map providers can be found. Therefore, it is necessary for the path planning algorithm to keep the USV a clearance distance from the islands to address the potential crowded traffic, shallow area problem and map inaccuracy problem.

2.3. Sea Current Effect

The general path planning approach finds the shortest path while avoiding obstacles whereas sea currents have a direct effect on the energy cost of the USV. The USV usually has limited energy and saving energy will improve the endurance of the mission execution. Downstream current will reduce energy consumption, however it may sacrifice the total distance. In some circumstances, it is also necessary to cross unflavoured surface current in order to save the energy cost in the whole journey. Therefore, it is necessary to develop an energy efficient path planning approach to take account of the sea current state.

Fig. 5. Illustration of data inaccuracy

3. Methodology

There are various ocean environmental disturbances that may have considerable effects on maritime vessels (Lee et al., 2011), i.e., current, wind, and waves. Thus, if the planned path does not consider these effects carefully, the forthcoming path following may be infeasible or even impossible. In addition, it might cost considerably more energy. Among these different environmental disturbances and uncertainties, sea current is the main focus of this paper.

For this purpose, this follow the approach (Lee et al., 2015): (1) the wind force is abstracted away by assuming there is no strong wind in the scenarios considered; (2) the wave effect cannot be ignored with respect to the magnitude of the forces, but we assume the frequency range does not affect the manoeuvrability of USVs, and the ocean wave only affects the floating body dynamics, which has more effect on dynamic positioning (DP)

rather than on Path Planning (PP).

Furthermore, the considered area is assumed to be a confined sea environment near a harbour with many small islands, and the total travelling distance is relatively short. Hence there is no dramatic change in environmental disturbances during the USVs mission. In this paper, sea current data is forecasted and updated hourly in most cases and usually does not change significantly during that period, it would be reasonable to assume the sea current state does not change during the mission.

This section is organized as follows: Section 3.1 describes the architecture of the proposed algorithm. Section 3.2 presents the environmental dataset used in this research, including the large spatial dataset and the sea current data. The Voronoi collision free roadmap generation is described in Section 3.3 and the Voronoi energy efficient path generation is presented in Section 3.4. Visibility graph generation approach is described in Section 3.5. Finally, the Voronoi-Visibility energy efficient path generation is introduced in Section 3.6.

3.1. Algorithm Architecture

The architecture of the proposed Voronoi-Visibility energy efficient path planning algorithm is shown in Fig. 6. This algorithm includes four parts: the collision free Voronoi roadmap generation, the Voronoi energy efficient path generation, the Visibility graph generation and Voronoi-Visibility energy efficient path generation.

In the collision free Voronoi roadmap generation, the coastline dataset is processed by the coastline expanding algorithm and the expanded coastlines will be used for keeping the subsequently generated roadmap a clearance distance c from the original coastlines. Then the expanded coastline is processed by the Voronoi diagram algorithm, the unreachable edges are removed and finally the collision free Voronoi roadmap is generated. In the Voronoi energy efficient path generation section, the starting point and the destination are inserted into the Voronoi roadmap. All the nodes of the Voronoi roadmap are stored in a node matrix. Then the energy consumption model is used to predict the energy cost of each edge of the Voronoi roadmap, and the corresponding energy cost weight is stored into the node matrix. The Dijkstra's search algorithm is applied to generate the Voronoi energy efficient path. In the Visibility graph generation section, the Voronoi energy efficient path is processed using Visibility graph that will provide more candidate paths into the Voronoi roadmap. Finally, the Dijkstra's search algorithm is applied again to search for the Voronoi-Visibility energy efficient path.

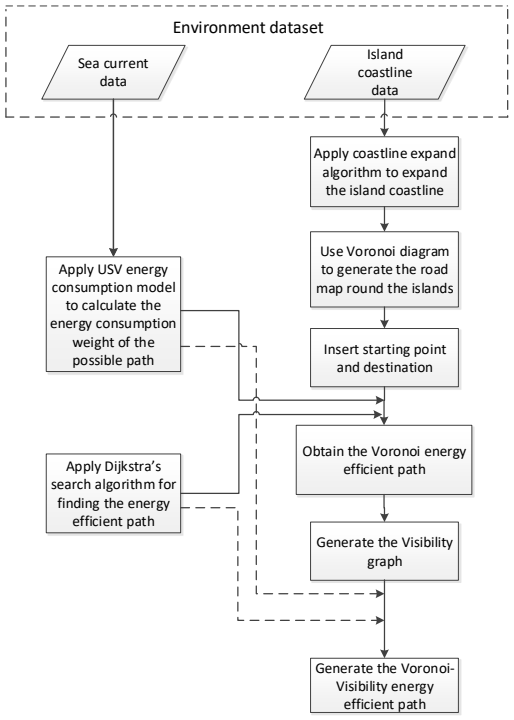


Fig. 6. The architecture of the proposed energy-efficient path planning algorithm

3.2. Environmental Dataset

The data used in the research include island coastline data and sea current data.

3.2.1. Coastline Data

In the USV mission simulation, real coastline data (i.e., the Global Self-Consistent Hierarchical High-Resolution Shorelines (GSHHS) dataset) was applied. The high-resolution coastline data used consist of points approximately 200 meters apart. The high-resolution real navigation data not only provide a real simulated environment but also can be used to demonstrate the efficient computing capability of the proposed algorithm.

3.2.2. Sea Current Data

The development of ocean science and satellite image processing techniques enabled the current ocean state to be predicted. The data used for the analysis in this paper were obtained from the company Tidetech Ltd (2016). The sea current data are compiled in grib files. The grib file format is a concise data format used to store historical and forecast weather data. The resolution, time step

and forecast length of the sea current data are provided in Table 1.

Table 1. Sea current data specifications.

Region	Parameters	Resolution (km)	Updating time step (hrs)	Forecast length (hrs)
North Atlantic	current	11	24	144
Gulf Stream	current	11	24	144
English Channel	current	2	0.25	48
North West Europe	current	20	1	120
Singapore Strait	tide current	0.8	1	48

The combination of the sea current data and the coastline data in the Singapore Strait is shown in Fig. 7.

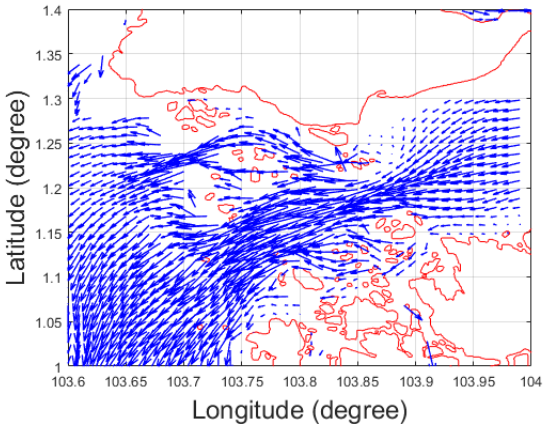


Fig. 7. Sea currents of the Singapore Strait: the current state was recorded at 5:00am, 11/06/2014. The red lines represent the coastlines of Singapore islands and the blue arrows show the sea current state.

3.3. Voronoi Roadmap Generation with Clearance c

The collision free Voronoi roadmap generation includes three steps: expanding the coastlines, implementing the Voronoi diagram algorithm and removing unreachable paths.

3.3.1. Coastline Expanding Algorithm

The raw coastline data should not be used directly because of the inaccuracy in the map data. To ensure the safety of the USV, each island coastline is extended

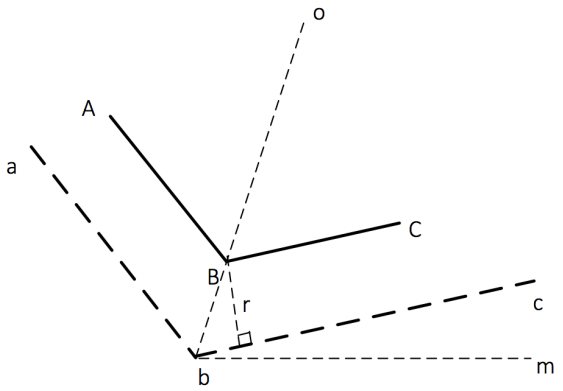


Fig. 8. Coastline expanding algorithm

by r meters, which is a configurable value based on the requirement of the user.

The coastline expanding algorithm is realised by calculating each expanded coastline point position. In Fig. 8, assuming coastline $A-B-C$ is expanded to $a-b-c$ by r meters, the segment $A-B$ is parallel to segment $a-b$ and their distance is r meters, and segment $B-C$ is parallel to segment $b-c$ and their distance is also r meters. The positions of points A, B and C are denoted by $A(x_A, y_A)$, $B(x_B, y_B)$ and $C(x_C, y_C)$.

The aim of the coastline expanding algorithm is to calculate the position of $b(x_b, y_b)$. The angle bisector of angle θ_{ABC} is denoted by line ob. The Line Of Sight (LOS) angle of line bc is denoted by θ_{cbm}. The LOS angle of line ba is denoted by θ_{abm}.

First, the angles θ_{abm} and θ_{cbm} are calculated using Eq. (1):

$$\theta_{abm} = atan(\frac{y_a - y_b}{x_a - x_b}),\ \theta_{cbm} = atan(\frac{y_c - y_b}{x_c - x_b}) \quad (1)$$

Then, the angle θ_{obc} and the length of Bb are calculated using Eq. (2):

$$\theta_{obc} = \frac{\theta_{abm} - \theta_{cbm}}{2},\ |Bb| = \frac{r}{sin(\theta_{obc})} \quad (2)$$

Next, the position of $b(x_b, y_b)$ can be calculated by Eq. (3) and Eq. (4):

$$x_b = x_B - |Bb| \times cos(\theta_{obc} + \theta_{cbm}) \quad (3)$$

$$y_b = y_B - |Bb| \times sin(\theta_{obc} + \theta_{cbm}) \quad (4)$$

Using the coastline expanding algorithm, all of the expanded coastline point positions can be calculated in

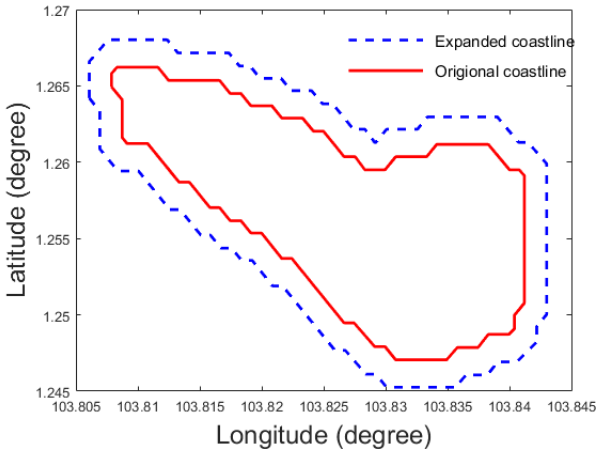


Fig. 9. Expanded island coastline

$O(n)$ time. Fig. 9 shows the result of expanding one of the Singapore strait islands. The red line is plotted using the original data from the GSHHS dataset. The blue line represents the expanded island coastline. The expanded distance is 200 meters.

All of the islands are expanded before implementation in the simulation. After extending the coastline, all of the candidate paths are checked to determine whether they cross the extended coastlines, and only the paths that avoid the coastline are retained to ensure that the USV is r meters away from the island. Therefore, the USV can travel safely with data inaccuracy.

3.3.2. Voronoi Diagram Generation

The construction of the Voronoi diagram of a set of obstacle vertices is both complex and time consuming. The Delaunay triangulation should be created first, and then generate the Voronoi diagram from it. The complexity of implementing Voronoi diagram is $O(nlog(n))$. For implementation, Delaunay triangulation generation and Voronoi diagram generation have been included in the built-in matlab function $[v, c] = voronoin(x, y)$. The inputs x and y represent the longitude and latitude of all of the expanded coastline points, correspondingly. The outputs v and c store the Voronoi nodes positions and the cell information, correspondingly. Then all of the vertices connection information are stored in a $N \times N$ matrix, which is called node matrix F. If node i and node j are found connected, then $F(i, j)$ and $F(j, i)$ of matrix equal 1, otherwise, $F(i, j)$ and $F(j, i)$ will equal 0. The node matrix F can be constructed by enquiring the node cell information, if the nodes are in the same cell and adjacent, we can say they are connected, otherwise, they are not connected. If in matrix F, only node i and node j are connected. The matrix F is shown in Table 2.

For demonstration, we applied Voronoi diagram approach to process the vertices of a polygon. In Fig. 10, the vertices of the polygon and the boundary points are used as the inputs of the Voronoi function. By processing the Voronoi node matrix and cell data, we can obtain the labelled Voronoi nodes. In Fig. 10, the adjacent nodes are connected using blue segments.

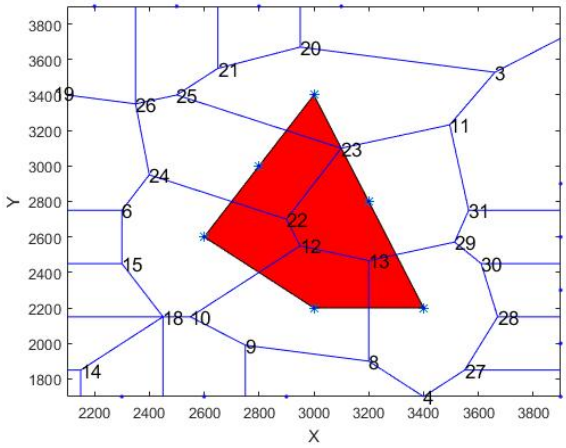


Fig. 10. The Voronoi diagram of a polygon

Table 2. Node matrix F

	...	i	...	j	...
...	0	0	0	0	0
i	0	0	0	1	0
...	0	0	0	0	0
j	0	1	0	0	0
...	0	0	0	0	0

The node matrix F stores the connectivity information of the Voronoi nodes. However, Fig. 10 indicates that not all of the blue lines are reachable for the USV because certain lines are inside the islands or across the coastlines. Therefore, all of the unreachable paths should be removed, and the node matrix should be modified accordingly.

3.3.3. Removing Unreachable Path

In Fig. 10, the segments 23-11, 23-25, 23-22, 22-24, 12-22, 12-10, 12-13, 13-8, and 13-29 are not reachable because they intersect with the obstacle. There are two kinds of paths required to be removed: Firstly, if

the path segment cross the coastlines, it needs to be removed. Secondly, if one node is inside the island profile, all the paths connected to this node need to be removed. For checking the clearance of one Voronoi edge, the complexity is $O(log(n))$. Because the Voronoi diagram has $O(n)$ edges, removing unreachable path requires $O(nlog(n))$ time. The node matrix F also needs to be modified correspondingly. Assume a removed path has two endpoints i and j, then the values of $F(i, j)$ and $F(j, i)$ will be updated to 0.

Multiple polygons and Singapore islands are applied for demonstration. In Fig. 11(a), the obstacles are represented by red polygons and the paths are represented by the blue segments. After the unreachable paths are removed, we get Fig. 11(b). Fig. 12(a) shows the Voronoi roadmap of the Singapore islands, with the blue lines representing the Voronoi roadmap. Because the data are high resolution, the road density is high. However, when the unreachable paths are removed, a clear roadmap emerges, which is shown in Fig. 12(b).

3.4. Voronoi Energy Efficient Path Generation

The Voronoi energy efficient path generation includes three steps: Firstly, inserting the starting point and the destination point into the Voronoi roadmap; Secondly, applying the energy consumption model to calculate the path segments weight; Thirdly, implementing Dijkstra's search algorithm to search the Voronoi energy efficient path from the starting point to the destination.

3.4.1. Insertion of Start Point and Destination

Assuming there are N nodes generated from the Voronoi diagram, then the nearest reachable Voronoi nodes to the start point and the destination point are node S and node D, respectively. The start point and the destination point are added to the Voronoi nodes as node $N + 1$ and node $N + 2$; The USV is then commanded to travel the path $N + 1...S...D...N + 2$. The node matrix F must be expanded to from size $N \times N$ to be size $(N + 2) \times (N + 2)$, as shown in Table 3.

Table 3. The node matrix F with start point and destination information

	...	S	D	...	N+1	N+2
...	...	0	0	...	0	0
S	0	0	0	0	1	0
D	0	0	0	0	0	1
...	...	0	0	...	0	0
N+1	0	1	0	0	0	0
N+2	0	0	1	0	0	0

3.4.2. USV Energy Consumption Function

The node matrix F is processed by Dijkstra's search algorithm to generate the most energy-efficient path. However, in matrix F, only 0's and 1's are included and the energy consumption weight information is not available for each Voronoi segment. In this section, the USV energy consumption model is applied to evaluate the energy requirements for each Voronoi segment.

For a USV that is commanded to travel from waypoint N_i to waypoint N_{i+1}, the ground speed of the USV is denoted by $\overrightarrow{v_g}$, the sea current speed is denoted by $\overrightarrow{v_c}$, and the relative USV speed is denoted by $\overrightarrow{v_u}$, which satisfy Eq. (5):

$$\overrightarrow{v_g} = \overrightarrow{v_c} + \overrightarrow{v_u} \tag{5}$$

The ship resistance is the most important term in determining the effective power of the given ship. Assuming the USV does not have the thrusters of the sway motion, only the surge speed relative to the water will be considered and the hydrodynamic drag F_g can be calculated by Eq. (6).

$$F_d = \frac{1}{2}\rho|v_u|^2 C_D A \tag{6}$$

Where ρ is the mess density of the water, C_D is the drag coefficient and A is the reference area. The USV energy consumption weight E can be calculated from the product of USV speed through water, drag force it experiences, and the travel duration. The USV energy consumption weight E can be calculated by using Eq. (7) and Eq. (8).

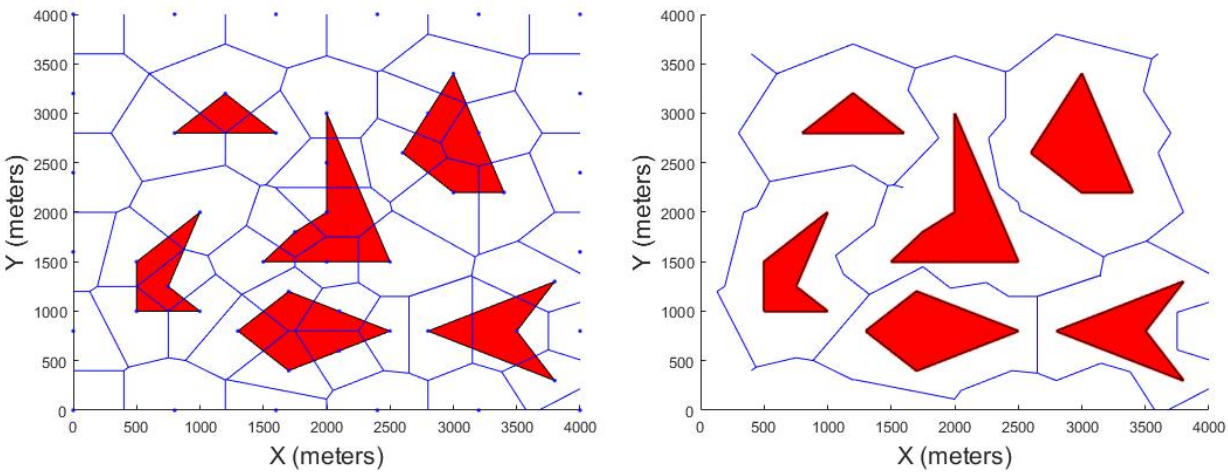
$$E = F_d \times |v_u| \times t \tag{7}$$

$$t = \frac{|\overline{N_i N_{i+1}}|}{|v_g|} \tag{8}$$

Where t is the time the USV consumed to travel from waypoint N_i to waypoint N_{i+1}. Then we can get Eq. (9).

$$E = \alpha|v_u|^3 \cdot \frac{|\overline{N_i N_{i+1}}|}{|v_g|} \tag{9}$$

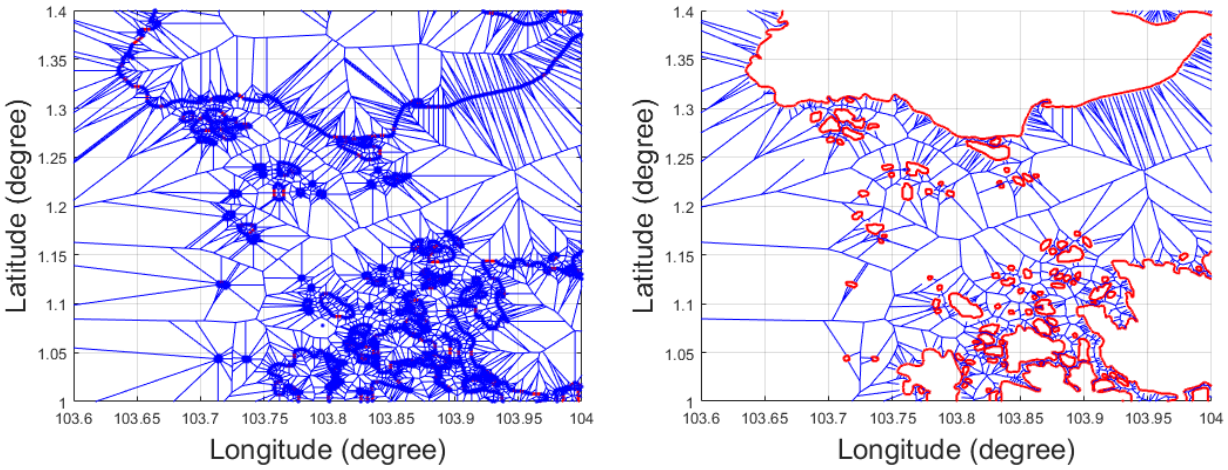
Where α is a combination of the water density, drag coefficient and the reference area, as α is a constant value, for simplicity, we assume α equals 1 in this research. Therefore, if the USV is commanded to travel at a constant ground speed $\overrightarrow{v_g}$, only the USV relative speed $\overrightarrow{v_u}$ and the waypoint segment length $|\overline{N_i N_{i+1}}|$ need to be calculated. Note that if the distance between two connected Voronoi nodes is longer than the scale of the sea



(a) Voronoi roadmap of polygon obstacles

(b) Collision free roadmap of polygon obstacles

Fig. 11. The Voronoi roadmap of polygon obstacles.



(a) The Voronoi roadmap of Singapore islands

(b) Collision free roadmap of Singapore islands

Fig. 12. The Voronoi roadmap of Singapore islands.

current, it will be divided into shorter segments before calculating the energy cost.

By processing the node matrix F, all of the corresponding matrix value of reachable segments are updated from 1 to their corresponding E matrix values. From node N_i to node N_j, the energy cost consumption weight is $E_{i,j}$, and from node N_j to node N_i, the energy cost consumption weight is $E_{j,i}$. Note that $E_{i,j}$ and $E_{j,i}$ do not equal each other because the sea current direction for travel segment N_iN_j is opposite with that of the segment N_jN_i. Therefore, a new node matrix is generated, as shown in Table 4.

Table 4. The node matrix F with energy consumption weight

	...	S	D	...	N+1	N+2
...	...	0	0	...	0	0
S	0	0	0	0	$E_{S,N+1}$	0
D	0	0	0	0	0	$E_{D,N+2}$
...	...	0	0	...	0	0
N+1	0	$E_{N+1,S}$	0	0	0	0
N+2	0	0	$E_{N+2,D}$	0	0	0

3.4.3. *Dijkstra's Search Algorithm*

By implementing Dijkstra's search algorithm to query the node matrix F, the energy efficient path can be generated. The pseudocode of the Dijkstra's search algorithm is given as in Fig. 13.

```
Function Dijkstra (Roadmap, start point)
    For each vertex A in Roadmap:              //initialization
        Energycost[A] :=infinity               // Initial energy cost from start point to vertex A
is set to infinite
        Previous[A] : = undefined              // Previous node in optimal path from start point
    Energycost[Start point] :=0                // Energy cost from start point to start point
    v := The set of all nodes in Roadmap       // All the nodes are stored in v
    While v is not empty:                      // main loop
        U := node in v with smallest Energycost[]
        Remove U from v
        For each neighbour A of U:             // where A has not yet been removed from v
            Alt := Energycost[U] + Energycost_between(U, A)
            if Alt < Energycost[A]
                Energycost[A] := Alt
                Previous[A] :=U
    Return previous[]
```

Fig. 13. The pseudocode of Dijkstra's search algorithm

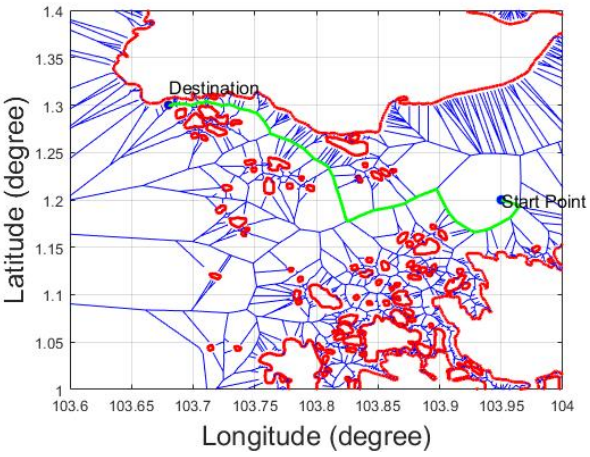


Fig. 14. Voronoi energy efficient path

A USV mission is used as an instance. The USV was commanded to travel from (103.95, 1.2) to (103.68, 1.3) at 05:00am on 11/06/2014. The USV kept a constant speed 1m/s. After the energy consumption model was implemented to calculate the corresponding energy consumption weight of node matrix, Dijkstra's search algorithm was implemented to search for the Voronoi energy efficient path, as shown in Fig. 14. The green line represents the Voronoi energy efficient path and the blue lines represent the Voronoi collision free roadmap. If we calculate the length of the connected nodes and implement Dijkstra's search algorithm, we can get the Voronoi shortest path. Note that the shortest path is referring to the shortest safe path in this paper. For comparison, the Voronoi shortest path is shown in Fig. 15. The black line represents the Voronoi shortest path. The Voronoi energy efficient path and shortest path are combined in Fig. 16. The sea current state is represented using blue arrows. The Voronoi energy efficient path is represented by green line and the Voronoi shortest path is represented by back line. We can see using the Voronoi energy efficient path will make the USV encounter more favourable sea current than the shortest path. However, the Voronoi paths are far from optimal. Therefore, Visibility graph algorithm is implemented to optimize the Voronoi path.

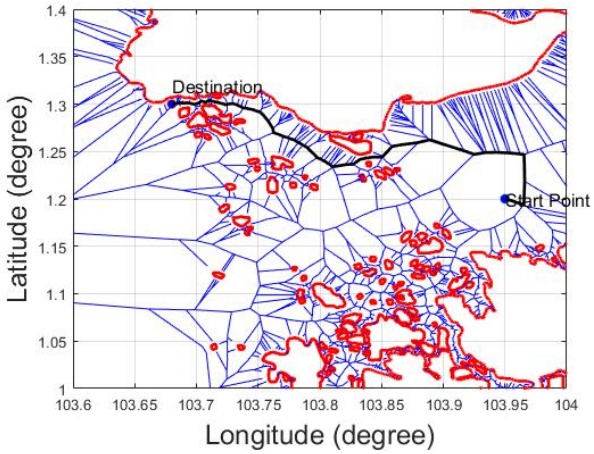


Fig. 15. Voronoi shortest path

3.5. *Visibility Graph Generation*

To refine the Voronoi energy efficient path, the Visibility graph algorithm is applied to provide more candidate paths to the Voronoi roadmap. Instead of applying the Visibility graph to all the vertices of the islands, we apply the Visibility graph to the Voronoi path obtained in last section. Assume for a waypoint w_i on the Voronoi energy efficient path ($i = 1..m$), where m is the number

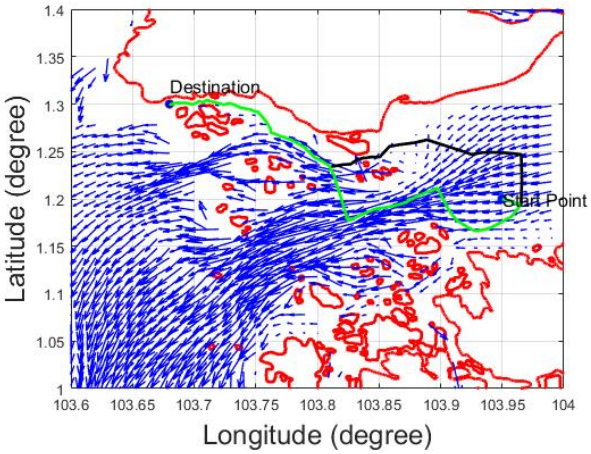


Fig. 16. Voronoi energy efficient path vs shortest path

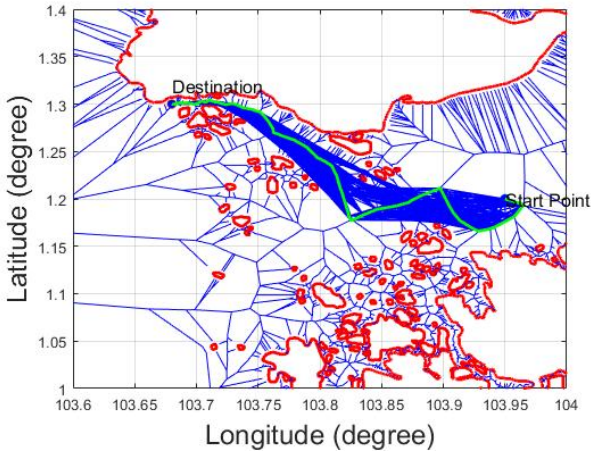


Fig. 18. Voronoi-Visbility roadmap

of the waypoints on the Voronoi path, we check whether the line segment $w_i w_1$ has intersection with the expanded coastlines. If there is no intersection, using the energy consumption model, we update the node matrix node $F(i,1) = E(i,1)$, otherwise, we check the line segment $w_i w_2$ until the line segment $w_i w_m$. The pseudocode is given in Fig. 17.

Function Visibility graph (VP, F)

Assume VP *is the Voronoi path, which has* m *waypoints.*

For $i = 1{:}m$

For $j = 1{:}m$

Check whether line segment $VP(i,j)$ *has intersection with the coastlines or not.*

If yes, continue.

If no, update $F(i,j) = E(i,j)$.

end

Fig. 17. The pseudocode of Visibility graph

Visibility graph has $O(m^2)$ edges and constructing the collision free Visibility graph requires $O(m^2 log(n))$ time. After applying the Visibility graph algorithm, we can get more candiate paths, as shown in Fig. 18. The Visibility graph is represented by the blue lines around the green line.

3.6. Voronoi-Visibility Energy Efficient Path Generation

The Dijkstra's search algorithm is applied again to search for the Voronoi-Visibility energy efficient path from the Voronoi-Visibility roadmap, as shown in Fig. 18. The generated Voronoi-Visibility energy efficient path is shown in Fig. 19, which is more smooth than the Voronoi energy efficient path generated in Section 3.4 and the path enables the USV follow the sea current direction more closely. For comparison, we also change the element of the node matrix F from the value of the energy consumption to the length of the segments, then we applied the Dijkstra's search algorithm, we get the Voronoi-Visibility shortest path. The Voronoi-Visibility energy efficient path and Voronoi-Visibility shortest path are combined in Fig. 20. The blue arrows represent the sea current state. The black line represents the Voronoi-Visibility shortest path and the green line represents the Voronoi-Visibility energy efficient path. It can be seen that the energy efficient path cost less energy by making the USV follow the direction of the favourable flow. The energy consumption of the Voronoi-Visibility energy efficient path is 0.1831 and the energy consumption of the shortest path is 0.2517, therefore, the energy efficient path saves 27.25% energy than the shortest path.

4. USV Missions Simulation

Singapore strait and Croatian islands were chosen as the mission area. The Singapore strait has 98 islands, whose coasline data are made of 4128 vertices, and it

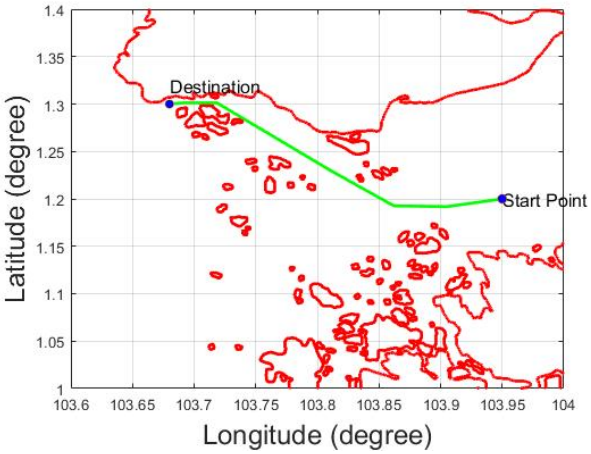


Fig. 19. Voronoi-Visibility energy efficient path

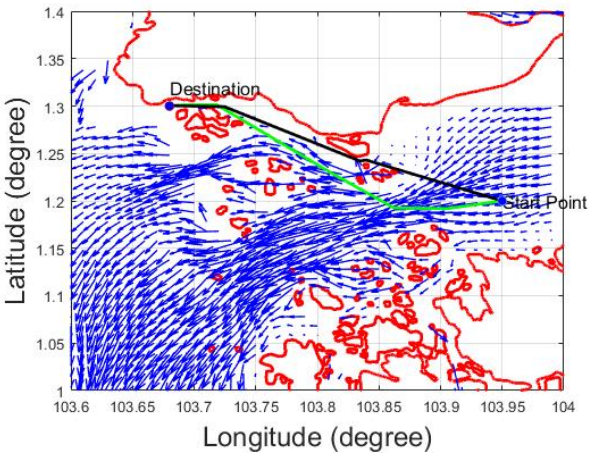


Fig. 20. Voronoi-Visibility energy efficient path vs shortest path: the green line represents the energy efficient path and the black line represents the shortest path

can be used for demonstrating the computational efficiency of proposed algorithm in dealing with the large spatial dataset. The Croatia missions were used for testing the performance of the proposed algorithm in different geographical scenario. Both the proposed energy-efficient path planning algorithm and the VV shortest path planning algorithm were tested in the ten Singapore missions and five Croatia missions. The effects of mission time, USV travelling speed and sea current states on the results were investigated. The computational time of the proposed Voronoi-Visibility algorithm was also compared with that of the Voronoi energy efficient path planning algorithm.

4.1. Visibility-Voronoi Shortest Path Planning Algorithm for Comparison

To demonstrate the energy efficiency of the proposed algorithm, the VV shortest path algorithm was developed for comparison purposes. In Fig. 21, the VV shortest path algorithm integrates the Voronoi algorithm, the Visibility algorithm and the Dijkstra's search algorithm. However, it only calculates the length of the connected nodes as the weight of the node matrix F. Then, the Dijsktra's search algorithm is implemented to calculate the shortest path from the start point to the destination. The VV shortest path algorithm architecture is provided in Fig. 21.

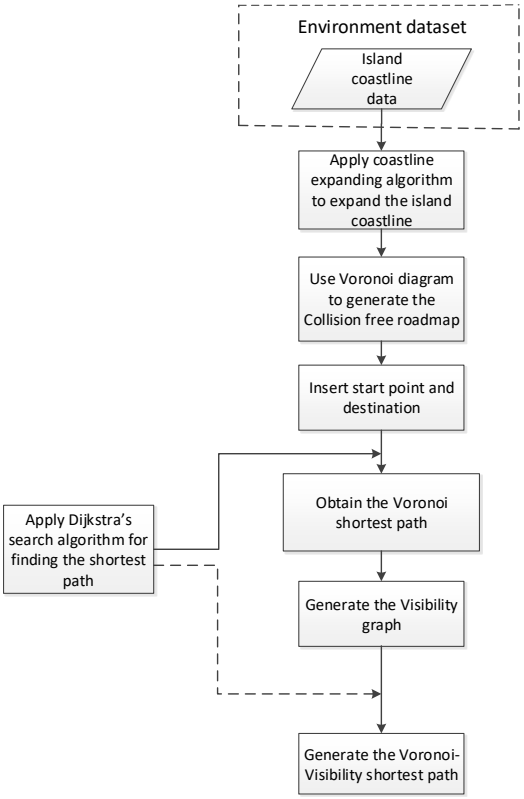


Fig. 21. The Voronoi-Visibility (VV) shortest path planning algorithm without the energy consumption model

4.2. Ten USV Mission Scenarios in Singapore Strait

Ten USV mission scenarios are used to demonstrate the energy efficiency of the proposed algorithm. The

date of these missions is 11/06/2014. These ten USV missions differ in starting point, destination, mission time and USV speed. The USV keeps constant speed while it is travelling and it is assumed when the USV is travelling, the sea current is spatially variant and not temporally variant. The comparison of energy efficient path and shortest path is given in Table 5. In these ten missions, the energy efficient paths require less energy than the shortest paths. The energy saving amount differs from 1.29% to 52.84%. The impacts of the mission time, travelling speed and the sea current state on the comparison results are given as below.

4.2.1. USV Missions with Different Mission Time

Missions No.1 and No.2 have different mission time. The corresponding energy efficient path and the shortest path are shown in Fig. 22 and Fig. 23. The green lines represent the energy efficient path and the black lines represent the shortest path. The blue arrows represent the direction and the magnitude of the sea current. Mission No.1 started at 10:00am, from Fig. 22, it can be found that the sea current flowed from east to west and the USV also travelled from east to west. Both the first half part and the second half part of the energy efficient paths made the USV take advantage of the sea current state. The shortest path made the USV cross the sea current in the beginning without taking account of the sea current state, therefore the energy efficient path saved more energy than the shortest path. In Mission No.2, the sea current state changed its direction and flowed from west to east, therefore, the energy efficient path was different with the path in Mission No.1. The first half of the energy efficient path made the USV cross the direction of the sea current instead of navigating the USV travel through the counter-flow directly like the shortest path.

The energy efficient path saved 46.48% energy than the shortest path in Mission No.1 and saved 15.08% in Mission No.2. It can be concluded that it is easier for the USV to save more percentage of the total energy than the corresponding shortest path when it travels from the east to the west in the morning than in the evening.

4.2.2. USV Missions with Different Travelling Speeds

Missions No.3, No.4 and No.5 have different travelling speeds and the path planning results are shown in Fig. 24, Fig. 25 and Fig. 26. The results show that the higher speed the USV utilizes, the more energy it will consume. In Mission No.3, the USV travelled at 1m/s, it consumed 0.0980 energy. However, in Mission No.5, travelling at 3m/s, it consumed 1.8301 energy. This is because travelling at a high speed will cause high hydrodynamic drag. The results also show that travelling at a low speed on the energy efficient path will save more percentage of total energy comparing with the corresponding shortest path than travelling at a high speed. In Mission No.3, it saved 52.84% of the total energy and in Mission No.5, it just saved 14.89% of the total energy.

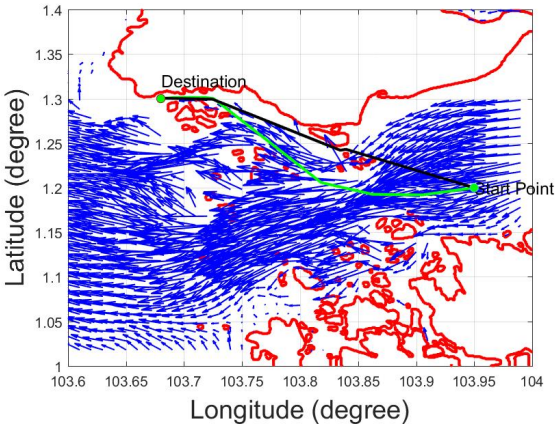


Fig. 22. Energy efficient path and shortest path in Mission No.1 (mission time: 10:00 am 11/06/2014)

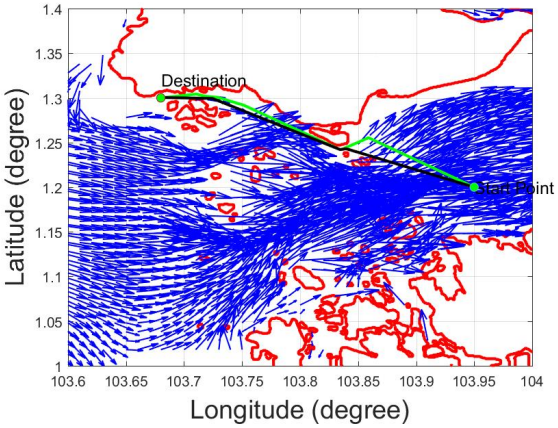


Fig. 23. Energy efficient path and shortest path in Mission No.2 (mission time: 18:00 pm 11/06/2014)

4.2.3. USV Missions in Favourable Flow, Cross-flow and Counter-flow

In Mission No.1, the USV encountered the favourable flow, as shown in Fig. 22. In Mission No.7, the USV en-

Table 5. Comparison of energy efficient path and shortest path in ten USV missions of Singapore Strait

No.	Start point	Destination	Mission time	Speed (m/s)	Efficient path length(km)	Shortest path length(km)	Efficient path energy cost	Shortest path energy cost	Energy saved
1	(103.95, 1.20)	(103.68, 1.30)	10:00	1	34.729	32.497	0.1157	0.2161	46.48%
2	(103.95, 1.20)	(103.90, 1.08	18:00	1	33.929	32.497	0.5068	0.5968	15.08%
3	(103.95, 1.20)	(103.75, 1.05)	06:00	1	31.910	28.647	0.0980	0.2079	52.84%
4	(103.95, 1.20)	(103.75, 1.05)	06:00	2	31.134	28.647	0.6868	0.9216	25.47%
5	(103.95, 1.20)	(103.75, 1.05)	06:00	3	31.052	28.647	1.8301	2.1503	14.89%
6	(103.95,1.15)	(103.75, 1.12)	06:00	1	26.123	23.054	0.0867	0.1305	33.59%
7	(103.68, 1.30)	(103.95, 1.20)	06:00	2	32.805	32.497	1.2592	1.2754	1.27%
8	(103.74, 1.30)	(103.80, 1.08)	06:00	2	25.558	25.509	0.8779	0.8955	1.97%
9	(103.98, 1.20)	(103.75, 1.05)	06:00	2	34.509	31.026	0.7815	0.9939	21.38%
10	(103.90, 1.08)	(103.68, 1.30)	06:00	2	36.690	36.361	1.1560	1.1713	1.30%

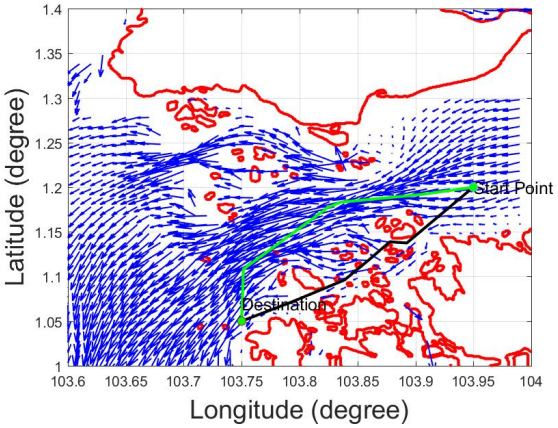


Fig. 24. Energy efficient path and shortest path in Mission No.3 (USV speed is 1m/s)

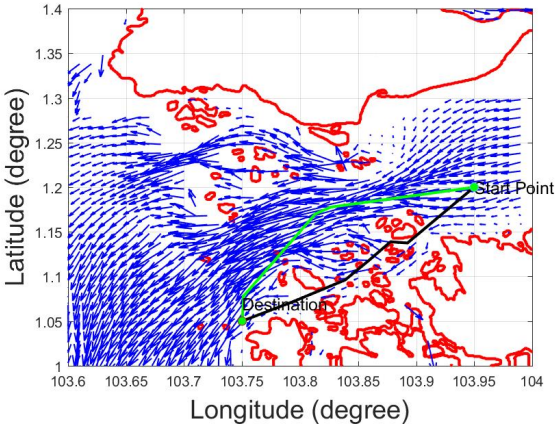


Fig. 25. Energy efficient path and shortest path in Mission No.4 (USV speed is 2m/s)

countered the counter-flow, as shown in Fig. 27. In Missions No.8 and No.10, the USV encountered cross flow, as shown in Fig. 28 and Fig. 29, respectively. In the favourable flow mission, the efficient path saved 46.48% energy. However, in the counter-flow and cross flow mission, the energy efficient paths are close to the corresponding shortest path and just saves small amount of energy comparing with the shortest path (saving 1.27% in Mission No.7, saving 1.97% in Mission No.8 and saving 1.30% in Mission No.10). This result indicates that in the favourable flow mission, it is easier for the proposed algorithm to save more percentage of total energy than the shortest path.

4.2.4. Computational Time

The path planning based on the Voronoi diagram has computational efficiency but the result is far from optimal. The proposed Voronoi-Visibility algorithm improves the quality of the Voronoi energy efficient path planning algorithm by using Visibility graph to generate more candidate paths. The Visibility graph can provide the optimal solution but the computational capability is not practical. The proposed algorithm intends to improve the quality of the Voronoi path and still keep the same level of the computational efficiency as that of the Voronoi energy efficient path planning algorithm.

The quality comparison of the Voronoi energy efficient path and the Voronoi-Visibility energy efficient path are shown in Fig. 30 and Fig. 31, correspondingly. The black line represents the Voronoi energy efficient path, the green line represents the Voronoi-Visibility energy efficient path and the blue arrows represent the sea current state. It can be seen that the Voronoi-Visibility path can navigate the USV to the area that has more favourable flow. Moreover, the Voronoi-Visibility path is more smooth than the Voronoi path.

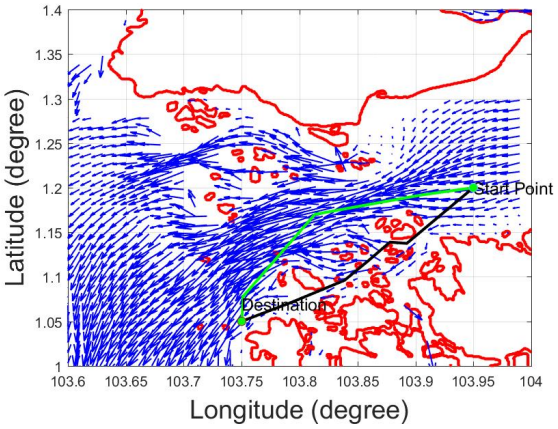


Fig. 26. Energy efficient path and shortest path in Mission No.5 (USV speed is 3m/s)

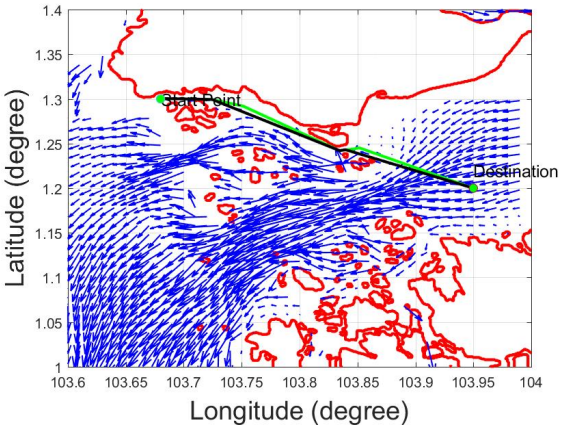


Fig. 27. Energy efficient path and shortest path in counter-flow situation (Mission No.7, sea current flows from east to west and the USV travels from west to east)

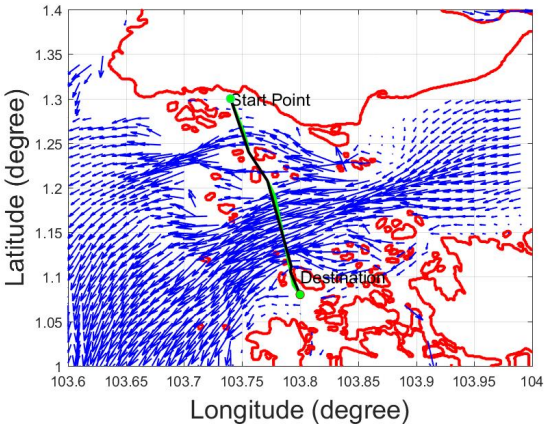


Fig. 28. Energy efficient path and shortest path in cross-flow situation (Mission No.8)

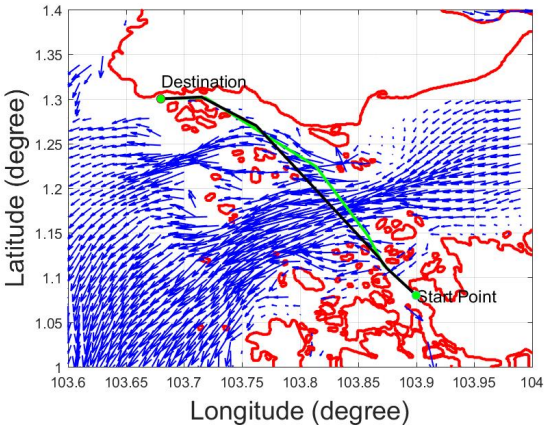


Fig. 29. Energy efficient path and shortest path in cross-flow situation (Mission No.10)

It is known that the Visibility graph can generate optimal result but will cost more computational time. Instead of applying Visibility graph algorithm to the islands vertices of the whole map, the proposed algorithm applies the Visibility graph algorithm partially to the waypoints of the Voronoi energy efficient path. The computational time of planning the ten missions is recorded, as shown in Table 6. The proposed Voronoi-Visibility energy efficient path planning algorithm is divided into four parts: collision free Voronoi roadmap generation (Part 1), Voronoi energy efficient path generation (Part 2), Visibility graph generation (Part 3) and Voronoi-Visibility energy efficient path generation (Part 4). t_V represents the total computational time of the Voronoi energy efficient path planning algorithm (Part 1 + Part 2), including collision free Voronoi roadmap generation and Voronoi energy efficient path generation. t_{VV} represents the total computational time of the whole proposed Voronoi-Visibility energy efficient path planning algorithm (Part 1 + Part 2 + Part 3 + Part 4). The waypoints number of the Voronoi energy efficient path, which is used for generating the Visibility graph, is also recorded. Note that all the experiments were carried out on a 2.7 GHz Intel Core i7-6820HK processor with 16.0 GB. The program was implemented in Matlab R2016b.

Table 6. Computational time of the Voronoi-Visibility path planning algorithm

No.	Part 1 (Seconds)	Part 2 (Seconds)	Part 3 (Seconds)	Part 4 (Seconds)	t_V (Seconds)	$t_{VV} - t_V$ (Seconds)	Number of waypoints on Voronoi path
1	24.648	1.189	2.132	0.116	25.837	2.248	97
2	24.467	1.188	2.825	0.119	25.655	2.944	113
3	26.861	1.412	0.787	0.114	28.273	0.901	51
4	25.965	1.288	0.761	0.119	27.253	0.880	51
5	23.277	1.178	0.734	0.119	24.455	0.853	51
6	26.234	1.648	0.421	0.117	27.882	0.539	34
7	25.036	1.388	2.402	0.118	26.424	2.520	101
8	28.485	1.276	1.084	0.116	29.761	1.200	66
9	23.773	1.308	0.794	0.120	25.081	0.915	52
10	25.989	1.234	3.036	0.118	27.223	3.154	115

* Part 1: Collision free Voronoi roadmap generation; Part 2: Voronoi energy efficient path generation; Part 3: Visibility graph generation; Part 4: Voronoi-Visibility energy efficient path generation; t_V: Part 1 + Part 2; t_{VV}: Part 1 + Part 2 + Part 3 + Part 4.

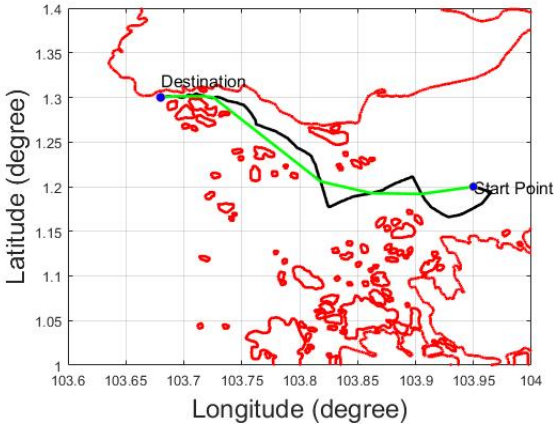


Fig. 30. Voronoi energy efficient path and Voronoi-Visibility energy efficient path in Mission No.1: The black line represents the Voronoi energy efficient path and the green line represents the Voronoi-Visibility energy efficient path

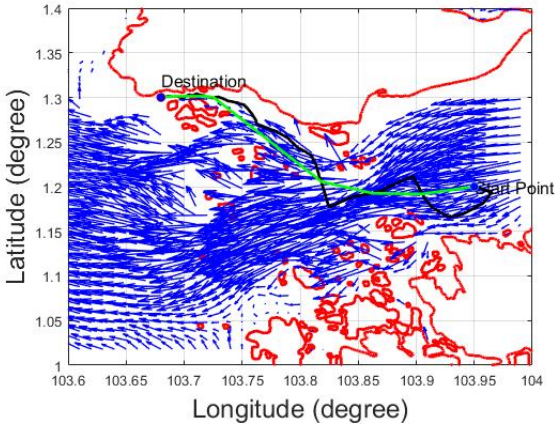


Fig. 31. Voronoi energy efficient path and Voronoi-Visibility energy efficient path with sea current state in Mission No.1

From Table 6, we can find the computational time of Voronoi energy efficient path planning algorithm (t_V) is between 24 and 30 seconds. The additional time of yielding the Visibility graph depends on the number of the waypoints on the Voronoi path those are used for building the Visibility graph (Part 3), which is between 0.421 seconds and 3.036 seconds. Building the Visibility graph (Part 3) even costs much less time than building the Voronoi collision free roadmap (Part 1). Comparing the computational time of the whole Voronoi-Visibility path planning algorithm (t_{VV}) and that of the Voronoi energy efficient path planning algorithm (t_V), we can say the Voronoi-Visibility algorithm still keeps the same level of the computational efficiency as that of the Voronoi algorithm. The reason is that we applied Visibility graph to the Voronoi path, whose waypoints number is between 34 and 115, instead of applying it to the entire map, whose vertices number is 4128. In our proposed algorithm, the collision free roadmap is constructed in $O((n + m^2)log(n))$ time and the number of the edges is $O(n + m^2)$. According to Bhattacharya and Gavrilova (2008), when applying Visibility graph algorithm to the islands with 1866 vertices and only the length of the segment was computed, the computing time was longer than 1 minute. In our case, if the Visibility graph algorithm is implemented to the whole map (4128 vertices), the computational time should be at minutes-level because $O(n^2)$ edges will be queried if

they intersect with the coastlines and the energy consumption cost also needs to be calculated for each edge. Through the quality comparison and the computing time comparison ($t_{VV} - t_V$), we can conclude that the proposed Voronoi-Visibility energy efficient path planning algorithm not only improves the quality of the Voronoi energy efficient path planning algorithm but also keeps the computational efficiency of the Voronoi energy efficient path planning algorithm, which makes it practical for processing large spatial dataset.

4.3. Five USV Mission Scenarios in Croatian islands

The proposed energy efficient algorithm and the shortest path planning algorithm are also compared in five mission scenarios of Croatian islands for testing the performance in different geographical scenario. The results are given in Table 7. It can be seen that the energy efficient path cost less energy than the shortest path, however, the amount of the saved energy rate is less than those of the Singapore missions. We analysed the Mission No.1 of Croatian islands and Mission No.4 of Singapore Strait, as shown in Fig. 32 and Fig. 33. The sea current of Croatia does not change as dramatically as Singapore area with the geographical variation. This leads to the small variation between the energy consumption amount of the energy efficient path and shortest path. However, the proposed energy efficient path planning algorithm can still provide energy efficient solutions in Croatia. In long range missions, the total energy saving amount cannot be ignored.

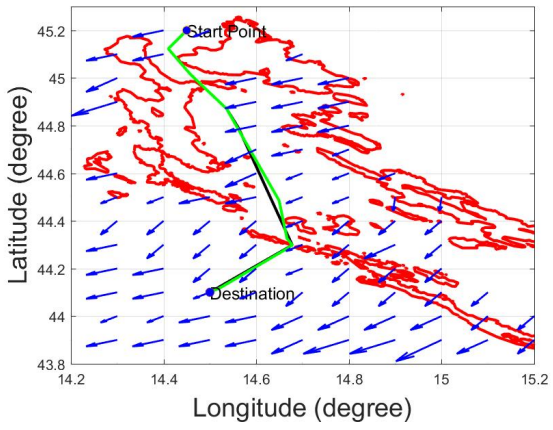


Fig. 32. Energy efficient path and shortest path in Mission No.1 of Croatian islands

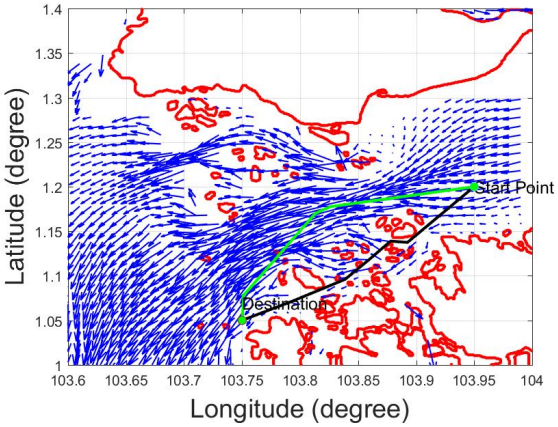


Fig. 33. Energy efficient path and shortest path in Mission No.4 of Singapore Strait

5. Conclusion and Future Work

In this research, the Voronoi-Visibility (VV) energy efficient path planning algorithm integrates the Voronoi diagram, Visibility graph, Dijkstra's search algorithm and energy consumption function. Moreover, by applying the coastline expanding algorithm, the proposed algorithm also ensures the USV safety by keeping the USV a user-configurable clearance distance from the coastlines. By searching for the candidate path with the minimum energy cost, the proposed algorithm yields the energy efficient path. The proposed algorithm intends to improve the quality of the path generated from the Voronoi diagram and still keeps the computational efficiency of the Voronoi path planning algorithm. The proposed VV energy efficient path and the VV shortest path were compared in ten missions of Singapore Strait. It can be found that the VV energy efficient path cost less energy than the shortest path, saving between 1.27% and 52.84%. By analysing the effects of the mission time, travelling speed and sea current states on the energy saving amount, we can conclude that in the favourable sea current and low speed situation, it is easier for the proposed VV energy efficient path to save more percentage of the total energy than the shortest path. Though the comparison of the quality and the computational time between the VV energy efficient path planning algorithm and Voronoi energy efficient path planning algorithm, we can find the proposed VV algorithm makes the path more smooth and enables the USV follow the favourable flow more closely, moreover, it still keep the same level of the computational efficiency as that of the Voronoi energy

Table 7. Comparison of energy efficient path and shortest path in five USV missions of Croatian islands

No.	Start point	Destination	Mission time	Speed (m/s)	Efficient path length(km)	Shortest path length(km)	Efficient path energy cost	Shortest path energy cost	Energy saved
1	(14.45, 45.20)	(14.50, 44.10)	03:00 a.m.	1	136.477	136.229	1.0561	1.0785	2.07%
2	(14.50, 44.80)	(14.50, 44.10)	03:00 a.m.	1	89.515	89.162	0.6338	0.6380	0.65%
3	(14.60, 44.90)	(14.50, 44.10)	03:00 a.m.	1	97.633	97.121	0.7052	0.7148	1.35%
4	(14.45, 45.20)	(14.50, 44.10)	06:00 a.m.	1	145.768	136.229	1.0094	1.0517	4.03%
5	(14.80, 44.50)	(14.80, 44.00)	06:00 a.m.	1	71.490	64.848	0.5046	0.5189	2.75%

efficient path planning algorithm, which demonstrates that the proposed Voronoi-Visibility integrates the advantages of the Voronoi diagram and Visibility graph. In the simulation of Croatian islands missions, the flexibility of using the proposed algorithm in different geographical scenario is demonstrated. It also shows that the proposed algorithm will save more energy in the fast changing sea current scenario than the regular scenario.

In the future work, the USV will be considered to be able to change speed according to the sea current state. In the long USV mission, the sea current data will also change with time. These two problems will increase the dimension of the path planning algorithm, which will increase the computational burden drastically. The A* algorithm (Zadeh et al., 2016), Mixed Integer Linear Programming (MILP)(Yilmaz et al., 2008), dynamic programming (Bryson, 1975) will suffer from expensive computational cost and criticized for their weak performance in high-dimensional problems. A new computational efficient searching algorithm should be developed to optimize the energy cost during the whole mission instead of replanning the path in each time interval (Song et al., 2017), which will only optimize the path locally. Apart from the path planning algorithm improvement, a decision making system should be developed to integrate the collision avoidance algorithm (Savvaris et al., 2014), path following algorithm (Niu et al., 2016a) and path planning algorithm (Niu et al., 2016b) to enable the USV to address more realistic scenarios. For a multiple USVs system, the decision making system that regulates the cooperation between the USVs will be critical and should be verified.

References

Agarwal, A., Lermusiaux, P. F., 2011. Statistical field estimation for complex coastal regions and archipelagos. Ocean Modelling 40 (2), 164–189.

Amato, N. M., Wu, Y., 1996. A randomized roadmap method for path and manipulation planning. In: Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on. Vol. 1. IEEE, pp. 113–120.

Avnaim, F., Boissonnat, J.-D., Faverjon, B., 1988. A practical exact motion planning algorithm for polygonal objects amidst polygonal obstacles. In: Robotics and Automation, 1988. Proceedings., 1988 IEEE International Conference on. IEEE, pp. 1656–1661.

Benavides, F., Tejera, G., Pedemonte, M., Casella, S., 2011. Real path planning based on genetic algorithm and voronoi diagrams. In: Robotics Symposium, 2011 IEEE IX Latin American and IEEE Colombian Conference on Automatic Control and Industry Applications (LARC). IEEE, pp. 1–6.

Bhattacharya, P., Gavrilova, M. L., 2008. Roadmap-based path planning-using the voronoi diagram for a clearance-based shortest path. IEEE Robotics & Automation Magazine 15 (2).

Bingham, B., Kraus, N., Howe, B., Freitag, L., Ball, K., Koski, P., Gallimore, E., November/December 2012. Passive and Active Acoustics Using an Autonomous Wave Glider. Journal of Field Robotics 29 (6), 911–923.

Breivik, M., Hovstein, V. E., Fossen, T. I., 2008. Straight-Line Target Tracking for Unmanned Surface Vehicles. Modeling, Identification and Control 29 (4), 131–149.

Bryson, A. E., 1975. Applied optimal control: optimization, estimation and control. CRC Press.

Candeloro, M., Lekkas, A. M., Sørensen, A. J., 2017. A voronoi-diagram-based dynamic path-planning system for underactuated marine vessels. Control Engineering Practice 61, 41–54.

Carpin, S., Pillonetto, G., 2005. Motion planning using adaptive random walks. IEEE Transactions on Robotics 21 (1), 129–136.

Garau, B., Alvarez, A., Oliver, G., 2005. Path planning of autonomous underwater vehicles in current fields with complex spatial variability: an a* approach. In: Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on. IEEE, pp. 194–198.

Ghosh, S. K., Mount, D. M., 1991. An output-sensitive algorithm for computing visibility graphs. SIAM Journal on Computing 20 (5), 888–910.

Hsu, D., Latombe, J.-C., Motwani, R., 1997. Path planning in expansive configuration spaces. In: Robotics and Automation, 1997. Proceedings., 1997 IEEE International Conference on. Vol. 3. IEEE, pp. 2719–2726.

Ibarra-Zannatha, J., Sossa-Azuela, J. H., Gonzalez-Hernandez, H., 1994. A new roadmap approach to automatic path planning for mobile robot navigation. In: Systems, Man, and Cybernetics, 1994. Humans, Information and Technology., 1994 IEEE International Conference on. Vol. 3. IEEE, pp. 2803–2808.

Kavraki, L. E., Latombe, J.-C., 1998. Probabilistic roadmaps for robot path planning.

Kim, J., Pearce, R. A., Amato, N. M., 2003. Extracting optimal paths from roadmaps for motion planning. In: Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on. Vol. 2. IEEE, pp. 2424–2429.

Koay, T.-B., Chitre, M., 2013. Energy-efficient path planning for fully propelled auvs in congested coastal waters. In: OCEANS-Bergen, 2013 MTS/IEEE. IEEE, pp. 1–9.

Konukoglu, E., Sermesant, M., Clatz, O., Peyrat, J.-M., Delingette, H., Ayache, N., 2007. A recursive anisotropic fast marching

approach to reaction diffusion equation: Application to tumor growth modeling. In: Information processing in medical imaging. Springer, pp. 687–699.

Kuffner, J., Latombe, J.-C., 2000. Interactive manipulation planning for animated characters. In: Computer Graphics and Applications, 2000. Proceedings. The Eighth Pacific Conference on. IEEE, pp. 417–418.

Kuffner, J. J., LaValle, S. M., 2000. Rrt-connect: An efficient approach to single-query path planning. In: Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on. Vol. 2. IEEE, pp. 995–1001.

Lee, T., Chung, H., Myung, H., 2011. Multi-resolution Path Planning for Marine Surface Vehicle considering Environmental Effects. In: Proceedings of OCEANS 2011 IEEE - Spain. IEEE, pp. 1–9.

Lee, T., Kim, H., Chung, H., Bang, Y., Myung, H., 2015. Energy efficient path planning for a marine surface vehicle considering heading angle. Ocean Engineering 107, 118–131.

Marbate, P., Jaini, P., 2013. Role of voronoi diagram approach in path planning. International Journal of Engineering Science and Technology 5 (3), 527.

Masehian, E., Amin-Naseri, M., 2004. A voronoi diagram-visibility graph-potential field compound algorithm for robot path planning. Journal of Field Robotics 21 (6), 275–300.

Mirebeau, J.-M., 2014. Anisotropic fast-marching on cartesian grids using lattice basis reduction. SIAM Journal on Numerical Analysis 52 (4), 1573–1599.

Naeem, W., Xu, T., Sutton, R., Tiano, A., June 2008. The design of a navigation, guidance, and control system for an unmanned surface vehicle for environmental monitoring. Proceedings of the Institution of Mechanical Engineers, Part M: Journal of Engineering for the Maritime Environment 222 (2), 67–79.

Niu, H., Lu, Y., Savvaris, A., Tsourdos, A., 2016a. Efficient path following algorithm for unmanned surface vehicle. In: OCEANS 2016-Shanghai. IEEE, pp. 1–7.

Niu, H., Lu, Y., Savvaris, A., Tsourdos, A., 2016b. Efficient path planning algorithms for unmanned surface vehicle. IFAC-PapersOnLine 49 (23), 121–126.

Noborio, H., Naniwa, T., Arimoto, S., 1990. A quadtree-based path-planning algorithm for a mobile robot. Journal of Field Robotics 7 (4), 555–574.

Pehlivanoglu, Y. V., 2012. A new vibrational genetic algorithm enhanced with a voronoi diagram for path planning of autonomous uav. Aerospace Science and Technology 16 (1), 47–55.

Petres, C., Pailhas, Y., Petillot, Y., Lane, D., 2005. Underwater path planing using fast marching algorithms. In: Oceans 2005-Europe. Vol. 2. IEEE, pp. 814–819.

Savvaris, A., Niu, H., A, O., Tsourdos, A., 2014. Development of collision avoidance algorithms for the c-enduro usv. IFAC Proceedings Volumes 47 (3), 12174–12181.

Song, R., Liu, Y., Bucknall, R., 2017. A multi-layered fast marching method for unmanned surface vehicle path planning in a time-variant maritime environment. Ocean Engineering 129, 301–317.

Sutton, R., Sharma, S., Xao, T., September 2011. Adaptive navigation systems for an unmanned surface vehicle. Journal of Marine Engineering & Technology 10 (3), 3–20.

Tidetech Ltd, 2016. Currents sample. [Online; accessed 12-Dec-2016].
URL `https://www.tidetech.org/data/`

Warren, C. W., 1989. Global path planning using artificial potential fields. In: Robotics and Automation, 1989. Proceedings., 1989 IEEE International Conference on. IEEE, pp. 316–321.

Wein, R., Van den Berg, J. P., Halperin, D., 2005. The visibility–voronoi complex and its applications. In: Proceedings of the twenty-first annual symposium on Computational geometry. ACM, pp. 63–72.

Yilmaz, N. K., Evangelinos, C., Lermusiaux, P. F., Patrikalakis, N. M., 2008. Path planning of autonomous underwater vehicles for adaptive sampling using mixed integer linear programming. IEEE Journal of Oceanic Engineering 33 (4), 522–537.

Zadeh, S. M., Powers, D. M., Sammut, K., Yazdani, A., 2016. Differential evolution for efficient auv path planning in time variant uncertain underwater environment. arXiv preprint arXiv:1604.02523.

2018-05-25

An energy-efficient path planning algorithm for unmanned surface vehicles

Niu, Hanlin

Elsevier

Niu H, Lu Y, Savvaris A, Tsourdos A. (2018) An energy-efficient path planning algorithm for unmanned surface vehicles. Ocean Engineering, Volume 161, August 2018, pp. 308-321
https://doi.org/10.1016/j.oceaneng.2018.01.025